SHLOK SHIVARAM IYER CLASS:3E THIRD SEMESTER CSE 1BM22CS260

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#include <stdlib.h>
#define MAX 100
char st[MAX];
int top=-1;
void push(char st[], char);
char pop(char st[]);
void InfixtoPostfix(char source[], char target[]);
int getPriority(char);
int main()
{
   char infix[100], postfix[100];

   printf("\n Enter any infix expression : ");
   scanf("%s",infix);

   InfixtoPostfix(infix, postfix);
   printf("\n The corresponding postfix expression is : ");
   puts(postfix);

   return 0;
}
void InfixtoPostfix(char source[], char target[])
{
   int i=0, j=0;
   char temp;
   strcpy(target, "");
   while(source[i]!='\0')
   {
      if(source[i]=='(')
      {
         push(st, source[i]);
         i++;
      }
      else if(source[i] == ')')
      {
         while((top!=-1) && (st[top]!='('))
         {
         target[j] = pop(st);
         j++;
         }
```

```
            if(top==-1)
            {
                printf("\n INCORRECT EXPRESSION");
                exit(0);
            }
            temp = pop(st);//remove left parenthesis
            i++;
        }
        else if(isdigit(source[i]) || isalpha(source[i]))
        {
            target[j] = source[i];
            j++;
            i++;
        }
        else if (source[i] == '+' || source[i] == '-' || source[i] == '*' ||
        source[i] == '/' || source[i] == '^')
        {
             while (top != -1 && st[top] != '(' && getPriority(st[top]) >= getPriority(source[i]))
            {
                target[j] = pop(st);
                j++;
            }
            push(st, source[i]);  // Push the current operator onto the stack
            i++;
        }
    }
    while((top!=-1) && (st[top]!='('))
    {
        target[j] = pop(st);
        j++;
    }
    target[j]='\0';
}
int getPriority(char op)
{

    if(op=='^')
        return(3);
    else if(op=='*' || op=='/')
        return(2);
    else if(op=='+' || op=='-')
        return(1);
    else
        return(0);
```

```
}

void push(char st[], char val)
{
   if(top==MAX-1)
      printf("\n STACK OVERFLOW");
   else
   {
      top++;
      st[top]=val;
   }
}
char pop(char st[])
{
   char val=' ';
   if(top==-1)
      printf("\n STACK UNDERFLOW");
   else
   {
      val=st[top];
      top--;
   }
   return val;
}
```

OUTPUT: